

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота №** **2**

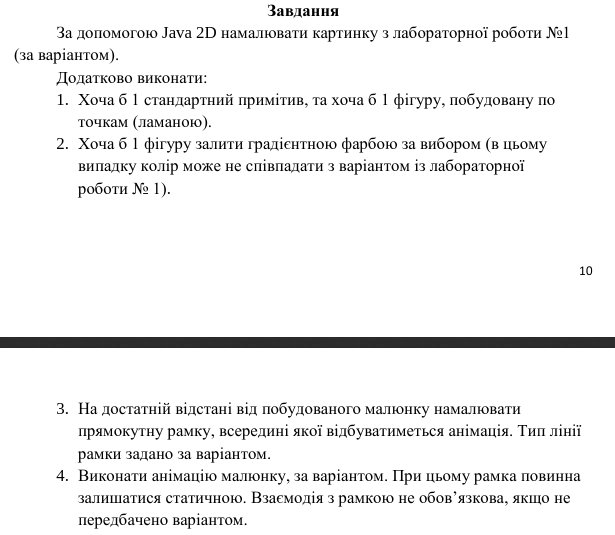
з дисципліни “Математичні та алгоритмічні основи комп’ютерної графіки”

|  |  |  |  |
| --- | --- | --- | --- |
| Виконав  студент III курсу  групи КП-81  Черняєв Іван  (*прізвище, ім’я, по батькові*)  варіант № 19 |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладачем  Шкурат Оксаною Сергіївною (*прізвище, ім’я, по батькові*) |

Київ 2021

**Варіант завдання:19**

**Завдання**:



**Лістинг коду програми**

package com.company;  
  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.awt.geom.GeneralPath;  
import javax.swing.JFrame;  
import javax.swing.JPanel;  
import javax.swing.Timer;  
  
  
public class Main extends JPanel implements ActionListener {  
  
 double points[][] = {  
 {-100, -15}, {-25, -25}, {0, -90}, {25, -25},  
 {100, -15}, {50, 25}, {60, 100}, {0, 50},  
 {-60, 100}, {-50, 25}, {-100, -15}  
 };  
  
 double width = 350;  
 double height = 210;  
  
 Timer timer;  
 // Для анімації повороту  
 private double angle = 0;  
  
 // Для анімації масштабування  
 private double scale = 1;  
 private double delta = 0.01;  
  
 // Для анімації руху  
 private double dx = 1;  
 private double tx = height / 2;  
 private double dy = 1;  
 private double ty = 0;  
 private static int *maxWidth*;  
 private static int *maxHeight*;  
  
 public Main() {  
 timer = new Timer(10, this);  
 timer.start();  
 }  
  
 public static void main(String[] args) {  
 JFrame frame = new JFrame("Lab2 ");  
 frame.add(new Main());  
 frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 frame.setSize(1000, 500);  
 frame.setResizable(false);  
 frame.setLocationRelativeTo(null);  
 frame.setVisible(true);  
  
 Dimension size = frame.getSize();  
 Insets insets = frame.getInsets();  
 *maxWidth* = size.width - insets.left - insets.right - 1;  
 *maxHeight* = size.height - insets.top - insets.bottom - 1;  
 }  
  
  
 public void paint(Graphics g) {  
 Graphics2D g2d = (Graphics2D) g;  
  
 Stroke defaultStroke = g2d.getStroke();  
 g2d.setStroke(defaultStroke);  
  
 g2d.setBackground(Color.*BLACK*);  
 g2d.clearRect(0, 0, *maxWidth* + 1, *maxHeight* + 1);  
  
 g2d.setRenderingHint(RenderingHints.*KEY\_ANTIALIASING*,  
 RenderingHints.*VALUE\_ANTIALIAS\_ON*);  
 g2d.setRenderingHint(RenderingHints.*KEY\_RENDERING*,  
 RenderingHints.*VALUE\_RENDER\_QUALITY*);  
  
 //рамка  
 drawBorder(g2d);  
 g2d.translate(*maxWidth* / 2, *maxHeight* / 2);  
  
 //aнімація  
 g2d.translate(tx, ty);  
 g2d.scale(scale, scale);  
  
 //малюнок  
 drawFigure(g2d);  
  
 }  
  
 private void drawBorder(Graphics2D g2d) {  
 g2d.setColor(Color.*YELLOW*);  
  
 BasicStroke basicStroke = new BasicStroke(5, BasicStroke.*CAP\_ROUND*, BasicStroke.*JOIN\_BEVEL*);  
 g2d.setStroke(basicStroke);  
  
 g2d.drawRect(5, 5, *maxWidth* - 10, *maxHeight* - 10);  
 }  
  
 public void actionPerformed(ActionEvent e) {  
 if (scale < 0.01) {  
 delta = -delta;  
 } else if (scale > 0.99) {  
 delta = -delta;  
 }  
  
 scale += delta;  
 angle += 0.03;  
  
 ty = height / 2 \* Math.*sin*(angle);  
 tx = height / 2 \* Math.*cos*(angle);  
  
 repaint();  
 }  
  
 private void drawFigure(Graphics2D g2d) {  
 GeneralPath bab = new GeneralPath();  
 double points2[][] = {  
 {width / 7 \* 2 - width / 2, height / 4.2 \* 0.6 - height / 2},  
 {width / 7 \* 2.8 - width / 2, height / 4.2 \* 1.5 - height / 2},  
 {width / 7 \* 5 - width / 2, height / 4.2 \* 1.5 - height / 2},  
 {width / 7 \* 5 - width / 2, height / 4.2 \* 2.3 - height / 2},  
 {width / 7 \* 2 - width / 2, height / 4.2 \* 2.3 - height / 2}};  
  
 bab.moveTo(points2[0][0], points2[0][1]);  
  
 for (int k = 1; k < points2.length; k++)  
 bab.lineTo(points2[k][0], points2[k][1]);  
 bab.closePath();  
  
 GradientPaint gp = new GradientPaint(5, 25,  
 Color.*YELLOW*, 20, 2, Color.*RED*, true);  
  
 g2d.setPaint(gp);  
 g2d.fill(bab);  
  
 g2d.setColor(new Color(117, 77, 0));  
 g2d.drawLine((int) (width / 7 \* 2.6 - width / 2), (int) (height / 4.2 \* 2.3 - height / 2), (int) (width / 7 \* 4 - width / 2), (int) (height / 4.2 \* 3.2 - height / 2));  
 g2d.drawLine((int) (width / 7 \* 4.4 - width / 2), (int) (height / 4.2 \* 2.3 - height / 2), (int) (width / 7 \* 3 - width / 2), (int) (height / 4.2 \* 3.2 - height / 2));  
  
 g2d.setColor(new Color(155, 102, 0));  
 g2d.drawLine((int) (width / 7 \* 5 - width / 2), (int) (height / 4.2 \* 1.5 - height / 2), (int) (width / 7 \* 5.65 - width / 2), (int) (height / 4.2 \* 0.8 - height / 2));  
  
 g2d.setColor(new Color(0.5411765F, 0.16862746F, 0.8862745F));  
 g2d.fillOval((int) (width / 7 \* 2.6 - width / 2) - (int) (width / 7 \* 0.44), (int) (height / 4.2 \* 3.4 - height / 2) - (int) (width / 7 \* 0.44), (int) (width / 7 \* 0.44) \* 2, (int) (width / 7 \* 0.44) \* 2);  
 g2d.fillOval((int) (width / 7 \* 4.4 - width / 2) - (int) (width / 7 \* 0.44), (int) (height / 4.2 \* 3.4 - height / 2) - (int) (width / 7 \* 0.44), (int) (width / 7 \* 0.44) \* 2, (int) (width / 7 \* 0.44) \* 2);  
  
 }  
}

**Результат**

